What is claimed is:

1. A method of speculative execution, comprising:
determining whether a mode is run-ahead execution or normal execution; and
upon a cache hit for a first cache line during run-ahead execution, setting a protection bit associated with the first cache line.

2. The method as in claim 1, further comprising:
upon a cache miss for a second cache line during run-ahead execution, evicting an unprotected cache line.

3. The method as in claim 2, further comprising:
upon a cache miss for the second cache line during run-ahead execution, replacing the evicted cache line with the second cache line and setting a protection bit associated with the second cache line.

4. The method as in claim 1, further comprising:
upon starting normal execution, clearing all protection bits.

5. The method as in claim 1, further comprising:
upon starting run-ahead execution, clearing all protection bits.

6. A method of replacing cache lines during run-ahead execution, comprising:
finding a potential victim in a cache;
determining whether a protection bit is set for the potential victim; and
evicting the potential victim only if the protection bit is clear.

7. The method as in claim 6, further comprising:
allocating a cache line into the cache to replace the potential victim; and

setting a protection bit associated with the allocated cache line.

8. The method as in claim 7, further comprising:

switching to normal execution;

referencing the allocated cache line; and

clearing the protection bit associated with the allocated cache line.

9. A method of accessing a cache, comprising:

determining whether a mode is run-ahead execution or normal execution; and

upon a cache miss during run-ahead execution, replacing a first cache line only if a protection bit associated with the first cache line is clear.

10. The method as in claim 9, further comprising:

upon a cache hit for a second cache line during run-ahead execution, setting a protection bit associated with the second cache line.

11. The method as in claim 9, further comprising:

upon a cache hit for a second cache line during normal execution, clearing a protection bit associated with the second cache line.

12. A method of executing a software prefetching thread on a multithreaded processor, comprising:

executing a software prefetching thread concurrently with normal threads in a program;

setting protection bits during execution of the software prefetching thread whenever cache lines are allocated and whenever there is a cache hit, the protection bits protecting cache lines from premature eviction; and clearing protection bits during execution of the normal threads as cache lines allocated for the software prefetching thread are referenced by the normal threads.

13. The method as in claim 12, further comprising:
clearing all protection bits when the software prefetching thread finishes executing.

14. The method as in claim 12, further comprising:
spawning the software prefetching thread for a predetermined section of code in the program.

15. The method as in claim 14, further comprising:
providing code for a software prefetching thread from an optimizing compiler.

16. A processor, comprising:
a cache having a plurality of cache lines;
a plurality of registers to store data for instructions to be executed by the processor;
circuitry to load data from the cache to the plurality of registers;
circuitry to prefetch data during speculative execution and to allocate cache lines to store the data; and
a plurality of identifiers associated with each cache line, each identifier to indicate whether to protect an associated cache line from premature eviction.

17. The processor as in claim 16, wherein
at least one of the plurality of identifiers to indicate whether the associated cache line is still in use.

18. The processor as in claim 16, wherein at least one of the plurality of identifiers to indicate whether the associated cache line was allocated during speculative execution and has yet to be touched during normal execution.

19. The processor as in claim 15, the cache further comprising:

a cache data memory; and

a cache directory to determine hits or misses and to store address tags of corresponding cache lines currently held in the cache data memory, the cache directory to store the identifiers.

20. The processor as in claim 15, the cache further comprising:

a cache controller to implement a cache strategy for moving data into and out of the cache data memory and the cache directory, the cache controller to store the identifiers.

21. A multiprocessor computer system, comprising:

a plurality of processors, each one of the processors having prefetcher logic and being capable of speculative execution;

at least one main memory;

at least one communication device coupling the plurality of processors to the at least one main memory;

a plurality of caches having a plurality of cache lines, each one of the plurality of caches associated with one of the plurality of processors; and

a protection bit associated with each of the cache lines in each of the plurality of caches, each protection bit to protect a cache line from premature eviction during speculative execution.

22. The multiprocessor computer system as in claim 21, further comprising:
control logic associated with the plurality of caches to manage the protection bits.

23. The multiprocessor computer system as in claim 22, further comprising:
at least one cache controller associated with the plurality of caches;
wherein the control logic resides in the at least one cache controller.

24. The multiprocessor computer system as in claim 21, further comprising:
a plurality of tag arrays associated with each cache;
wherein the protection bits reside in each tag array associated with each cache.

25. A computer system , comprising:
a main memory;
a processor;
a bus to connect the main memory and the processor;
a cache associated with the processor, the cache having a plurality of cache lines; and
a protection bit associated with each of the cache lines in each of the plurality of caches, each protection bit to protect a cache line from premature eviction during speculative execution.

26. The computer system as in claim 25, wherein
the cache is a level one (L1) cache.

27. The computer system as in claim 26, wherein
the level one (L1) cache is on the same chip die as the processor.

28. The computer system as in claim 25, wherein

the cache is a level two (L2) cache.